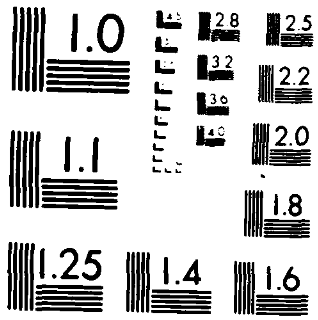
AD-A124 087 ACCURACY AND COMPLETENESS OF PROBLEM SOLUTIONS WITH EXAMPLE-SOLUTIONS(U) PERFORMANCE MEASUREMENT ASSOCIATES INC VIENNA VA E M CONNELLY NOV 82 PMA-82-363 1/1

UNCLASSIFIED N00014-79-C-0739 F/G 9/2 NL

END
DATE
FILMED
2-83
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963-A

(12)

PMA

12

FINAL
REPORT

Edward M. Connelly

ACCURACY AND
COMPLETENESS
OF PROBLEM
SOLUTIONS WITH
EXAMPLE
SOLUTIONS

This research is supported
by Engineering Psychology
Group, Office of Naval Research.

December 1982

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER 82-363	2. GOVT ACCESSION NO. AD-A124 037 / 3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) ACCURACY AND COMPLETENESS OF PROBLEM SOLUTIONS WITH EXAMPLE-SOLUTIONS	5. TYPE OF REPORT & PERIOD COVERED Final Report / 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Edward M. Connelly	8. CONTRACT OR GRANT NUMBER(s) N00014-7[illegible]-C-07[illegible]
9. PERFORMING ORGANIZATION NAME AND ADDRESS Performance Measurement Associates, Inc. 410 Pine Street, S.E. Vienna, Virginia 22180	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS [illegible] 42; RR 042-0[illegible], RR 042-06-01; NR [illegible]
11. CONTROLLING OFFICE NAME AND ADDRESS Engineering Psychology Group Office of Naval Research 800 N. Quincy Street, Arlington, Va. 22217	12. REPORT DATE November 1982 / 13. NUMBER OF PAGES 14
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) Department of the Navy Office of Naval Research Arlington, Virginia 22217	15. SECURITY CLASS. (of this report) Unclassified / 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for Public Release. Distribution Unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

None

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Software Development, Programmer Behavior, Automatic Programming, Automatic Processor

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This research investigated the ability of computer users, both programmers and non-programmers, to specify problem solutions in the form of example-solutions. This ability was evaluated as a function of the complexity of the processor, i.e. the degree of generalization of the user inputs, the complexity of the problem, and the complexity of the feedback-aids. The experimental task employed in this study required

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

the specification of a logic for the formation of a naval task-force. The performance both of programmers and non-programmers decreased with increasing levels of problem-complexity and with reduced processor support. For both the groups, errors-of-commission were relatively infrequent compared to errors-of-omission. It was found that the degree of processor-complexity was much more influential than problem-complexity in predicting performance scores. When computer generalization of user-input was provided, performance was significantly lower than during all other experimental conditions. Results also showed that participant-strategy in the generation of problem solutions was a significant factor in performance, though years-of-experience and years-of-education were not found to be good predictors of performance. The feedback-aids were shown to be most effective when they included the logic implied by the example-solutions. These experiments demonstrate the effectiveness of the on-line use of computer software to create and modify software routines.

TABLE OF CONTENTS

ABSTRACT

This research investigated the ability of computer users, both programmers and non-programmers, to specify problem solutions in the form of example-solutions. This ability was evaluated as a function of the complexity of the processor, i.e. the degree of generalization of the user inputs, the complexity of the problem, and the complexity of the feedback-aids. The experimental task employed in this study required the specification of a logic for the formation of a naval task-force. The performance both of programmers and non-programmers decreased with increasing levels of problem-complexity and with reduced processor-support. For both the groups, errors-of-commission were relatively infrequent compared to errors-of-omission. It was found that the degree of processor-complexity was much more influential than problem-complexity in predicting performance scores. When little computer generalization of user-input was provided, performance was significantly lower than during all other experimental conditions. Results also showed that participant-strategy in the generation of problem solutions was a significant factor in performance, though years-of-experience and years-of-education were not found to be good predictors of performance. The feedback-aids were shown to be most effective when they included the logic implied by the example-solutions. These experiments demonstrate the effectiveness of the on-line use of computer software to create and modify software routines.

INTRODUCTION

This is the final report on Contract N00014-79-C-0739, Work Unit Number N[illegible]-101 between Performance Measurement Associates, Inc. and the Engineering Psychology Group, Office of Naval Research. The contract was initiated on 1 Sept 1979 and ended on [illegible] 1982.

The research investigated the capability of programmers and non-programmers to specify problem solutions by developing example-solutions and also by writing computer programs; each method of specification was accomplished at various levels of problem-complexity. The level of difficulty of each problem was reflected by the number of steps needed by the user to develop a solution. Machine processing of the user-inputs permitted inferences to be developed about the algorithms required to solve a particular problem. The iterative feedback of processing results led users to a more precise definition of the desired solution.

Six experiments were conducted, with the same problems used in all experiments. The ability of the participants to develop example-solutions was evaluated as a function of the participants' background and experience, the complexity of the problem to be solved, the level of processing provided by the computer, and the level of feedback-aids, when aids were available.

Two technical reports were published and three papers were presented documenting the experiments and results obtained. These documents are identified in the list of technical reports [illegible] at the end of this report.

Experiments 1 and 2 were designed to investigate the ability of expert programmers and of bookkeepers/accountants who were not expert programmers to develop example-solutions for a hypothetical Navy task-force problem. The experimental variables for both experiments were problem-complexity and processing complexity, i.e. the amount of machine processing of [illegible].

Experiments 3 and 4 were designed to investigate the ability of expert programmers and non-programmers to develop accurate and complete example-solutions using various feedback-

aids at various levels of problem-complexity. The four [illegible] designs were based on the results of Experiment [illegible], when the systematic generation of example-solutions, [illegible] a combinational-measure, had been shown to [illegible] correlated with performance (explaining 60% of [illegible] variance).

Experiment 5 was designed [illegible] of expert programmers to revise [illegible] the form of example-solutions [illegible] incorrect entries had been introduced, [illegible] in Experiments 3 and 4.

Finally, Experiment 6 [illegible] to develop computer code written in [illegible] levels of data input – a design [illegible] design of Experiment 1. The results [illegible] sub-routines written in FORTRAN IV [illegible] a ship combination, as that combination [illegible].

The performance measures used in [illegible] consisted of error-measures and strategies-measures. [illegible] error-measures were:

a. P_T, the probability that a given ship-combination was correctly classified as acceptable or unacceptable.

b. P_C, the probability that a correct [illegible] combination was accepted.

c. P_{IC}, the probability that an incorrect ship-combination was rejected.

In addition to the error-measures above, relative error-measures were used. A relative error-measure was defined as a participant's error-score (P_T, P_C, P_{IC}) on an experimental problem minus his/her error-score on the pre-test problem. The relative error-measures thus tended to remove the effect of the participant's innate capability, and, as a result, were more sensitive to experiment factors than were the error-measures alone.

Two strategy-measures were used to detect the frequency with which participants used specific strategies. One strategy-measure, the combinational-measure, detected the frequency with which a participant changed only one component at a time of [illegible] example-solution. Another strategy-measure, the sequence-measure, detected the use-patterns of the various feedback-aids.

RESULTS OF EXPERIMENTS

Experiments 1 and 2

The results of Experiments 1 and 2, [illegible] categories, are described below. The design of Experiments 3 through 6 was largely based on them.

Processor Complexity and [illegible]

First, as expected, more errors occurred [illegible] work on the more complex problems. However, the level of processing, or generalization, of the example-solutions was found to be an important error-reducing factor, i.e., a significant reduction in errors occurred when data from example-solutions were processed into a standard form and presented to the participant.

Systematic Strategies and Feedback-Aids

A second result, and perhaps the most important, was that participants in both categories who performed well tended to use a systematic, step-by-step strategy in selecting example-solutions. This result, together with the first, noted above, suggested that feedback-aids might be designed to encourage participants to use systematic strategies, by processing their example-solutions and then feeding back the resultant data to suggest possible additional inputs.

Breadth vs. Depth of Experience

A third result of the first two experiments applied to the subsequent experiments was that the number of years of advanced education (i.e., beyond high school) and the number of years

of professional experience were found to be unimportant [illegible] in predicting performance. As a consequence of [illegible], additional demographic factors were evaluated for the [illegible] in the subsequent experiments, in an effort to reveal [illegible] predictors of performance.

Low Frequency of Errors-of-Comm[illegible]

The fourth result applied to [illegible] was the observation that only a few errors-of-comm[illegible] during the generation of the example-[illegible]. [illegible] errors that did occur were errors-of-[illegible]. [illegible] result influenced the design of Experi[illegible], [illegible] code was written to solve the same [illegible] 1, so that a comparison of error-[illegible].

Experiment [illegible]

Feedback-Aids

Three aids were developed. Aid #1 provided the [illegible] selection logic (SSL) implied by the participant's example-solutions. Aid #2 included the SSL and an ordered listing [illegible]-type of the example-solutions input by the participant. The ordered listing was intended to aid the participant by showing possible omissions in ship-combinations. Aid #3 included the SSL and a list of suggested ship-combinations to consider. The suggested ship-combinations were those logically required to complete the combinations suggested by the example-solutions previously entered.

The effect of the feedback-aids as measured by the error-score (P_C), i.e. by the probability of accepting a correct ship-combination, was not statistically significant. However, the effect of the aids as measured by the relative error-score (error-score on the experiment problem minus the error-score on the pretest problem) was found to be significant and important. Apparently, the feedback-aids did help at least a portion of the participant population -- the less-than-superior performers. Those who would perform well without the aids were not helped by the aids. Also, it was apparent that variations in performance due to the participants' innate abilities were greater than variations in performance due to the feedback-aids. This factor,

plus the observation that superior performers may not need or use the aids, may account for the insignificant effect of the aids on error-score and the significant effect on relative-[illegible]-score.

Feedback-Aid #6 appeared to affect performance to a greater degree than Aid #2. Aid #6 included recommendations of logical example-solutions based on [illegible] example-solutions input previously. Aid #2, on the other hand, included an ordered list of example-solutions previously [illegible]. With Aid #2, the participant had to examine the pattern of previous inputs and identify any missing example-solutions. With Aid #6, the participant was presented with recommendations. [illegible] example-solutions previously [illegible] not in error, [illegible] reasonable to expect that Aid #6 might support performance [illegible] than that supported by Aid #2.

Aid #3 could give false recommendations, however, without indicating the basis for these recommendations. If, for instance, Aid #6 was used early in the problem, when only a few example-solutions had been input by the participant, there were possible next-logical patterns which were not correct. Also, when the participant inputed an example-solution that contained an error, Aid #6 would recommend solutions that actually completed the error-induced patterns. Such aids as Aid #6 can thus be either helpful or harmful, depending on how they are used. When recommended solutions are used without careful evaluation, this type of feedback-aid is potentially harmful. If, however, the display of patterns built upon an error increases the likelihood of the error's detection - by displaying its impact - then aids that provide recommended solutions may help the user to detect input errors.

Breadth vs. Depth of Experience

The lack of a strong predictive relationship between years-of-higher-education or years-of-experience and performance may come as a surprise to educators and directors of personnel departments. This result was found in all of the experiments, so that very strong evidence is available to support the assertion that years-of-education and relevant work-experience are not good predictors of problem-solving performance.

Additional results suggest that the "number of programming languages (used on 1 or more programs)" and "number of operating systems used" are better predictors of the [illegible] of computer users/programmers.

Combinational Strategy

Combinational-strategy was [illegible] predictor of performance. [illegible] Experiments 1 and 2, where approximately the [illegible] variance was explained by combinational-strategy [illegible] Experiments 1 and 2 and [illegible] in Experiments [illegible] and 4). [illegible] the combinational-[illegible] to-moment measure [illegible] operational-[illegible]. A moment-to-moment measure [illegible] lated to performance and thus provided greatly improved [illegible] sensitivity.

Experiment 5

Feedback-Aids

In Experiment 5, participants were asked to [illegible] example-solutions that included various numbers of [illegible] combinations. This experiment design permitted calculation [illegible] the probability of maintaining an initially-correct example-[illegible] and the probability of detecting and correcting an initially-[illegible] example-solution.

Analyses of grand-mean probabilities of success [illegible] that the probability of maintaining an initially-correct solution was .93. However, there was a probability of .65 for detecting and correcting an erroneous example-solution. Obviously, there was a performance decrement in the detection and correction of erroneous example-solutions. Useful feedback -aids might thus be directed toward the detection and correction of existing errors.

Feedback-Aids #2 and #3, which were the same as those used in Experiments 3 and 4, were not, however, a benefit to participants engaged in the revision of example-solutions. Further, an analysis using relative measures, which tended to remove the effect of participant skill, did not result in statistically significant results. The conclusion, then, was that the strategies used

successfully to revise existing solutions. Results, however, [illegible] that the aids designed to help in the development of new solutions were not helpful in revising incorrect problem-solutions.

Relativity of Error Detection

But there appears to have been a more [illegible] in detection of errors. As the number of incorrect example-solutions was decreased, there was a decrease in the [illegible] detecting and correcting these incorrect example-solutions. The fewer the number of errors, the lower was the probability of detecting a given error.

This result suggests that there may be a base-line probability (based on the frequency of errors recently detected) of detecting and correcting errors which affects the probability of judging that any solution is incorrect. Thus, the "correctness" of a solution as determined by a user may be a function of:

1. The actual correctness of that solution,

2. The perceived frequency of erroneous solutions found recently.

This hypothesis, consistent with predictions of signal detection theory (which says that the probability of an event is a function of a base-line probability in addition to specific measurements on the signal itself), predicts a decreasing probability of detecting and correcting errors with decreasing error-rates. This means that the probability of detecting the last few errors may be so small that it is not likely that they will be found. But it also suggests a possible solution: seeding errors (which can later be removed if not detected) to increase the base-line error rate and, thus, to increase the probability of detecting unknown errors.

Experiment 6

Two types of errors were analyzed. One type, termed an "error-of-omission", referred to an error that resulted in a failure to accept a correct entity (e.g.ship combination). When specifying a problem solution with example-solutions, an error-

of-omission could be [illegible] to enter [illegible] of a suitable entity ([illegible]). [illegible] second type of error considered was an "error-of-[illegible]." [illegible] example-solutions were used to [illegible], an error-of-commission corresponded to [illegible] processor which was [illegible] example. An error-of-[illegible] accepting incorrect [illegible].

<u>Errors-of-Omission</u>

There was [illegible] complexity on error-of-[illegible] of specifying problem [illegible], i.e., [illegible] FORTRAN IV subroutine[illegible].

<u>Errors-of-Commission</u>

When [illegible] example-[illegible] aids, the rate of error-of-com[illegible] problem complexity-[illegible] Halstead's E Metric.* But, given [illegible], such as in Experiment 3, [illegible] could be eliminated, as evidenced [illegible] which performance degradation [illegible].

The most important [illegible] errors-of-commission was that specification by example-solution was [illegible] to specification by program code. Analysis of the mean [illegible] from Experiments 1, 2, and 3 provided [illegible] evidence [illegible] example-solutions substantially reduced errors-of-commission compared to using FORTRAN IV program code. The 3% rate for errors-of-commission with example-solutions compared favorably with 18% for program code.

Three hypotheses concerning the superior performance of the example-solution method seem plausible:

1. It was working with examples and dealing with each individual combination of items one-at-a time that resulted in a low rate of errors-of-commission.

*For a discussion of Halstead's E Metric, See Connelly, Comeau & Johnson, Technical Report 81-361, 1981.

2. It was the specification of each solution one-at-a-time that alone was [illegible]. In other words, if computer programs were developed [illegible] each solution combination one-at-a-time, the rate of errors-of-commission would [illegible].

3. The [illegible] of the example-solution [illegible] was due, in part, to the [illegible] of example-solutions from one [illegible] into another form, [illegible] different form, [illegible], it was [illegible] that [illegible] the user to view [illegible] way and that resulted in a low [illegible] errors-of-commission. Consequently, if [illegible] by the user were transformed into a different logic form and fed back to the user for approval, a low rate of errors-of-commission would be obtained.

These hypotheses are not alternative hypotheses – all could be true. We have strong evidence that the first hypothesis is true. If the second is true but not the third, program-design and coding methods could be adapted to a more combination-dependent structure. And finally, if the third hypothesis were found to be true, pre-compilation [illegible] the user's program code into another form (while maintaining the same program logic) for feedback to the user.

CONCLUSIONS

1. Feedback-aids, to support use of example-solutions, should include the logic implied by the example-solutions as well as new example-solutions that complete the logic patterns suggested by the existing set of example-solutions.

2. Feedback-aids consisting of an ordered listing of all present solutions also supported high performance; but, the predictive aid type referred to above is preferred.

3. Feedback-aids assisted [illegible] the performance of both programmers and non-programmers who did not perform well without [illegible].

4. Performance measures designed to detect feedback performance improvement with feedback-aids should be relative measures, which indicate the difference between performance on a common task and on an experimental task.

5. The lack of a strong relationship between "years-of-higher-education", "years-of-experience" and performance, coupled with the strong relationship between "number of computer languages" known and "number of operating systems" used, suggests that education and experience should not be used as they have been in the past for hiring, promotion, determining salary level, and assigning tasks. Instead, the number of computer languages known and the number of operating systems used, which are better performance predictors, should be used until the underlying factors [illegible] are discovered.

6. Apparently, the depth of an individual's experience is not as important to performance as is the breadth of his experience.

7. A possible common underlying experience-related factor is the ability to view problems from alternative viewpoints, or the ability to develop alternative approaches to problems – an ability that might be enhanced with feedback-aids.

8. The performance-prediction capability of strategy-measures, developed as moment-to-moment measures, not only clearly demonstrates that systematic strategies were used by successful participants (which led to the design of the feedback aids), but also convincingly demonstrates that moment-to-moment measures provide the sensitivity to explain considerable performance variance (approximately 60% in Experiments 1 thru 4).

9. When modifying initially-incorrect example-solutions, the probability (.96) of maintaining an initially-correct example-solution was approximately the same as the probability of developing a new correct example-solution. Also, the mean probability of detecting and correcting an erroneous example-solution was low (.[illegible]). It is suggested that new aids should be developed to [illegible] in the detection and correction of existing errors.

10. The probability of detecting and correcting an erroneous example-solution decreased as the initial number of erroneous example-solutions was decreased. The fewer the total number of errors the less was the likelihood of detecting a given error. This suggests that a method for increasing the probability of detecting errors is to seed errors, unknown to the individual reviewing the example-solutions (or computer program) but otherwise recorded, to increase the base-line rate of error detection and therefore to increase the probability of detecting an unknown error.

11. Software error-categories are typically defined only to facilitate data collection and recording. However, analysis of software errors showed that when an error category was decomposed into sub-categories the independent variables in the prediction equations changed. It is concluded, therefore, that software error-categories should be selected with regard to predictability as well as to data collectability. Two sub-categories should be combined only when their prediction equations are homologous, i.e. have the same independent variables.

12. The superior performance (fewer errors-of-commission) achieved when using example-solutions and inductive processing to specify problem solutions over the performance achieved when using FORTRAN IV code may provide a basis for determining the underlying mechanism for that success and a means for incorporating that

mechanism into program-design- and coding-aids. Apparently, superior performance was [illegible] either because each combination of the input variables was treated individually and/or because the example-solutions were transformed into another logic form -- the [illegible] selection logic ([illegible]). If the former is a significant factor, then [illegible] should be adapted to program-design- and coding-aids. If the latter is a significant factor then [illegible] design- and coding-aids should be developed to transform the logic provided by the user into another form which is then fed back to the user for his review. Such a transformation might present the program's equivalent logic.

RECOMMENDATIONS FOR FURTHER RESEARCH

1. The strategy-measures used to analyze FORTRAN IV were not moment-to-moment measures. Instead, they were a classification of types of possible strategies. The predictive power of the measures was only moderate compared to those used to evaluate performance in developing example-solutions. It is suggested that moment-to-moment strategy-measures be developed for both program-design and program-code tasks.

2. Feedback-aids designed to support development of original example-solutions were not found applicable to the revision of erroneous example-solutions. Since use of example-solutions is a viable way to specify problem solutions in itself, and reveals ways of improving program design and code, successful revision-strategies should be identified, and revision-aids should be derived from those strategies.

3. The "number of programming languages" known and the "number of operating systems" used have been shown to be good predictors of performance both in developing example-solutions and in writing program code. It is also suggested that the ability to develop alternative approaches may be a common

factor which may be enhanced by learning new languages and operating systems, and which may be a key performance-factor. The underlying factor or factors resulting in superior performance should be determined to assist in personnel selection and design of improved aids.

4. It is suggested that error-detection capability may be a function of a base-line error-[illegible] rate, and therefore that seeding errors unknown to the individual checking the material may increase the probability of detecting an unseeded error. This conjecture should be subjected to experimental test to determine if seeding improves performance, and, if so, what frequency and type of seeded errors should be used.

5. The basis for superior performance with example-solutions needs to be resolved. The concept of writing program code for each combination of factors (or the use of an aid to automatically analyze logic to help develop accurate combination-independent logic) and the concept of code transformation into a different logical form for feedback to the user for approval need to be contrasted in an experimental environment. There is a potential here for substantially increasing the correctness of computer programs if an aid can be developed to permit transfer of the superior, almost error-free performance with example-solutions to code-writing performance.

6. Independent of the methods for improving performance in writing program code suggested in Recommendation 5, a new aid design should be considered that combines general statements written in program code with redundant example solutions -- i.e. with example-solutions that are not part of a program test, but, instead, that are inductively transformed into an alternative code. A pre-compiler aid would produce actual code from both sources. Potential performance-improvement could provide high-quality, almost error-free code.

ACKNOWLEDGEMENTS

The author is grateful to Robert F. Cen[illegible] who con-tributed significantly to the development of the in[illegible] con[illegible] and who wrote the system and analysis programs for [illegible]. His interest and enthusiasm for the work contributed [illegible] success of the project. The author is also grateful to [illegible] Johnson who, while with us for only a short time, contributed positively by conducting Experiments #6 and #4 and by providing data analysis for those experiments. Further, the author is grateful to Mike Olshausen for dedication to his task of [illegible] throughout all revisions. Finally, the author is grateful to Dr. John J. O'Hare and Dr. Martin [illegible] for their [illegible] interest in the effort.

TECHNICAL REPORTS

Connelly, E. M., [illegible], [illegible], [illegible] automatic [illegible] for computer [illegible]. [illegible] Performance Measurement Associates, [illegible]. AD A108[illegible]70

Connelly, E. M. A [illegible] of problem solution [illegible] program code. (Technical Report [illegible]). [illegible] Measurement Associates, September [illegible].

TECHNICAL PAPERS

Connelly, E. M., & [illegible], [illegible] increasing levels of [illegible] presented at the [illegible] Annual [illegible] of the Washington, D.C. [illegible] A.M., University of Maryland, College Park, Maryland, [illegible].

Connelly, E. M., & [illegible], [illegible] problem complexity on [illegible] solutions for computer [illegible]. Paper presented at the Human Factors Society [illegible] Annual Conference, Rochester, New York, October [illegible].

Connelly, E. M. The effect of [illegible] complexity, computer power, and feedback [illegible] on the accuracy and completeness of computer programs. Paper presented at the Human Factors Society 26th Annual Conference, Seattle, Washington, October 1982.

DISTRIBUTION LIST

OSD

CAPT Paul R. Chatelier
Office of the Deputy Under Secretary
of Defense
OUSDRE (E&LS)
Pentagon, Room 3D129
Washington, D.C. 20301

Department of the Navy

Engineering Psychology Programs
Code 442
Office of Naval Research
800 North Quincy Street
Arlington, VA 22217 (5 cys)

Electronics & Electromagnetics
Technology Programs
Code 250
Office of Naval Research
800 North Quincy Street
Arlington, VA 22217

Communication & Computer Technology
Programs
Code 240
Office of Naval Research
800 North Quincy Street
Arlington, VA 22217

Tactical Development & Evaluation
Support Programs
Code 230
Office of Naval Research
800 North Quincy Street
Arlington, VA 22217

Department of the Navy

Manpower, Personnel and Training
Programs
Code 270
Office of Naval Research
800 North Quincy Street
Arlington, VA 22217

Physiology & Neuro Biology Programs
Code 441B
Office of Naval Research
800 North Quincy Street
Arlington, VA 22217

Special Assistant for Marine
Corps Matters
Code 100M
Office of Naval Research
800 North Quincy Street
Arlington, VA 22217

Commanding Officer
ONREAST Office
ATTN: Dr. J. Lester
Barnes Building
495 Summer Street
Boston, MA 02210

Commanding Officer
ONRWEST Office
ATTN: Dr. E. Gloye
1030 East Green Street
Pasadena, CA 91106

Office of Naval Research
Scientific Liaison Group
American Embassy, Room A-407
APO San Francisco, CA 96503

Department of the Navy

Director
Naval Research Laboratory
Technical Information Division
Code 2627
Washington, D.C. 20375

Dr. Michael Melich
Communications Sciences Division
Code 7500
Naval Research Laboratory
Washington, D.C. 20375

Dr. L. Chmura
Code 7592
Naval Research Laboratory
Washington, D.C. 20375

Dr. Robert G. Smith
Office of the Chief of Naval
Operations, OP987H
Personnel Logistics Plans
Washington, D.C. 20350

CDR G. Worthington
Office of the Chief of Naval
Operations, OP-372G
Washington, D.C. 20350

Dr. W. Mehuron
Office of the Chief of Naval
Operations, OP 987
Washington, D.C. 20350

Dr. Andrew Rechnitzer
Office of the Chief of Naval
Operations, OP 952F
Naval Oceanography Division
Washington, D.C. 20350

Dr. Jerry C. Lamb
Combat Control Systems
Naval Underwater Systems Center
Newport, RI 02840

Department of the Navy

Naval Training Equipment Center
ATTN: Technical Library
Orlando, FL 32813

Human Factors Department
Code N-71
Naval Training Equipment Center
Orlando, FL 32813

Dr. Alfred F. Smode
Training Analysis and Evaluation
Group
Naval Training Equipment Center
Code TAEG
Orlando, FL 32813

CDR Norman E. Lane
Code N-7A
Naval Training Equipment Center
Orlando, FL 32813

Dr. R. J. K. Jacob
Code 7590
Naval Research Laboratory
Washington, D.C. 20375

Dr. Gary Poock
Operations Research Department
Naval Postgraduate School
Monterey, CA 93940

Dean of Research Administration
Naval Postgraduate School
Monterey, CA 93940

Mr. Warren Lewis
Human Engineering Branch
Code 8231
Naval Ocean Systems Center
San Diego, CA 92152

Information Sciences Division
Code 433
Office of Naval Research
800 North Quincy Street
Arlington, VA 22217

Department of the Navy

Dr. A. L. Slafkosky
Scientific Advisor
Commandant of the Marine Corps
Code RD-1
Washington, D.C. 20380

Dr. John Impagliazzo
Code 101
Naval Underwater Systems Center
Newport, RI 02840

Naval Material Command
NAVMAT 0722 - Rm. 508
800 North Quincy Street
Arlington, VA 22217

Commander
Naval Air Systems Command
Human Factors Programs
NAVAIR 340F
Washington, D.C. 20361

Commander
Naval Air Systems Command
Crew Station Design
NAVAIR 5313
Washington, D.C. 20361

Mr. Phillip Andrews
Naval Sea Systems Command
NAVSEA 0341
Washington, D.C. 20362

Commander
Naval Electronics Systems Command
Human Factors Engineering Branch
Code 81323
Washington, D.C. 20360

Dr. George Moeller
Human Factors Engineering Branch
Submarine Medical Research Lab
Naval Submarine Base
Groton, CT 06340

Department of the Navy

Head
Aerospace Psychology Department
Code [illegible]
Naval Aerospace Medical Research Lab
Pensacola, FL [illegible]

Commanding Officer
Naval Health Research Center
San Diego, CA [illegible]

Dr. James McGrath
CINCLANT [illegible]
Code 004 [illegible]
Norfolk, VA [illegible]

Navy Personnel Research and
Development Center
Planning & Appraisal Division
San Diego, CA [illegible]

Dr. Robert Blanchard
Navy Personnel Research and
Development Center
Command and Support Systems
San Diego, CA [illegible]

Mr. Stephen Merriman
Human Factors Engineering Division
Naval Air Development Center
Warminster, PA 18974

Dr. [illegible] Hopson
Human Factors Engineering Division
Naval Air Development Center
Warminster, PA 18974

Mr. Jeffrey Grossman
Human Factors Branch
Code 315[illegible]
Naval Weapons Center
China Lake, CA 93555

Department of the Navy

Human Factors Engineering Branch
Code 1226
Pacific Missile Test Center
Point Mugu, CA 93042

Mr. J. Williams
Department of Environmental
Sciences
U.S. Naval Academy
Annapolis, MD 21402

Dean of the Academic Department
U.S. Naval Academy
Annapolis, MD 21402

Dr. S. Schiflett
Human Factors Section
Systems Engineering Test
Directorate
U.S. Naval Air Test Center
Patuxent River, MD 20670

Department of the Army

Mr. J. Barber
HQS, Department of the Army
DAPE-MBR
Washington, D.C. 20310

Technical Director
U.S. Army Research Institute
5001 Eisenhower Avenue
Alexandria, VA 22333

Director, Organizations and
Systems Research Laboratory
U.S. Army Research Institute
5001 Eisenhower Avenue
Alexandria, VA 22333

Technical Director
U.S. Army Human Engineering Labs
Aberdeen Proving Ground, MD 21005

Department of the Air Force

U.S. Air Force Office of Scientific
Research
Life Sciences Directorate, NL
Bolling Air Force Base
Washington, D.C. [illegible]

Chief, Systems Engineering Branch
Human Engineering Division
USAF AMRL/HES
Wright-Patterson AFB, OH [illegible]

Dr. Earl Alluisi
Chief Scientist
AFHRL/CCN
Brooks AFB, TX 78235

Foreign Addressees

Dr. Kenneth Gardner
Applied Psychology Unit
Admiralty Marine Technology
Establishment
Teddington, Middlesex TW11 OLN
England

Director, Human Factors Wing
Defense & Civil Institute of
Environmental Medicine
Post Office Box 2000
Downsview, Ontario M3M 3B9
Canada

Dr. A. D. Baddeley
Director, Applied Psychology Unit
Medical Research Council
15 Chaucer Road
Cambridge, MA CB2 2EF
England

Other Government Agencies

Defense Technical Information Center
Cameron Station, Bldg. 5
Alexandria, VA 22314 (12 cys)

Other Government Agencies

Dr. Craig Fields
Director, System Sciences Office
Defense Advanced Research Projects Agency
1400 Wilson Blvd
Arlington, VA 22209

Dr. M. Montemerlo
Human Factors & Simulation Technology, RTE-6
NASA HQS
Washington, D.C. 20546

Other Organizations

Dr. H. McI. Parsons
Human Resources Research Office
300 N. Washington Street
Alexandria, VA 22314

Dr. Jesse Orlansky
Institute for Defense Analyses
1801 N. Beauregard Street
Alexandria, VA 22311

Dr. Robert T. Hennessy
NAS - National Research Council (COHF)
2101 Constitution Ave., N.W.
Washington, D.C. 20418

Dr. Robert Williges
Dept of Industrial Engineering & OR
Virginia Polytechnic Institute and State University
130 Whittemore Hall
Blacksburg, VA 24061

Dr. Deborah Boehm-Davis
General Electric Company
Information Systems Programs
1755 Jefferson Davis Highway
Arlington, VA 22202

Other Organizations

Dr. Edward R. Jones
Chief, Human Factors Engineering
McDonnell-Douglas Astronautics Company
St. Louis Division
Box 516
St. Louis, MO 63166

Dr. Richard Pew
Bolt Beranek & Newman, Inc.
50 Moulton Street
Cambridge, MA 02238

Dr. David J. Getty
Bolt Beranek & Newman, Inc.
50 Moulton Street
Cambridge, MA 02238

LMED

-83